Testing Requirements to Manage Data Exchange Specifications in Enterprise Integration – A Schema Design Quality Focus

Boonserm (Serm) Kulvatunyou
Nenad Ivezic
Buhwan Jeong

Manufacturing Systems Integration Division, National Institute of Standards & Technology
Gaithersburg, MD 20899-8260, U.S.A

Abstract

In this paper, we describe the requirements to test W3C XML Schema usage when defining message schemas for data exchange in any large and evolving enterprise integration project. We then decompose the XML Schema testing into four (4) aspects including the message schema conformance to the XML Schema specification grammar, the message schema conformance to the XML Schema specification semantics, the message schema conformance to design quality testing, and canonical semantics testing of the message schema. We describe these four testing aspects in some detail and point to other related efforts. We further focus to provide some technical details for the message schema design quality testing. As a future work, we describe the requirements for canonical semantics testing and potential solution approaches. Finally, we describe an implementation architecture for the message schema design quality testing.

Keywords: Enterprise Integration, XML Schema Design, Metadata Management

1. Overview

Large enterprise integration projects are typically evolving and distributed in nature as they involve tens or even hundreds of various applications that, in turn, may translate into thousands of application-to-application connections.

These integration projects are evolving because it is impossible (1) to obtain sufficient resources to complete the project in one budget cycle, (2) to identify all the integration needs at the beginning of the project and (3) to complete the entire project in a short period from the management perspective. These projects are also distributed because (1) software applications are distributed geographically across the enterprise and (2) domain experts and implementation teams are distributed geographically also.

Such evolving and distributed characteristics of any large integration project cause potential long-term interoperability problems within the project if it is not managed appropriately. The interoperability problems can delay the project completion and significantly increase the project costs [NIST1]. Such project characteristics further necessitate that any large integration project should include a coordination authority responsible for ensuring consistency among integration subgroups. Such a coordination authority may need to keep oversight over the integration architecture, integration technologies, workflow/processes, and data exchange specifications (including information models). This paper focuses on the data exchange consistency, which often takes the most time and cost in an integration project [Linthicum 1999].

Traditionally, the data exchange consistency is achieved via long discussions and meetings among domain experts. However, participants within a coordination authority team may or may not be domain experts. In addition, the team typically does not have the necessary capability as well as the capacity to oversee all the application areas. More importantly, the integration subprojects involve several heterogeneous domains. The domain experts in the coordination authority team lose their time when the team reviews subprojects that are irrelevant to their expertise. The traditional approach alone that resolves the inconsistency by gathering all experts results in significant waste; hence, innovative measures and tools are necessary to assist the coordination authority team. In this paper, we propose some tools to help reduce the time, cost, and loss, while maximizing consistency and interoperability during the integration project life cycle.

2. Components in Data Exchange Specifications Consistency

At present, the W3C XML Schema [W3C 2001] is widely used for integration projects as a canonical representation for data exchange specifications. Therefore, our work is centered on the XML Schema usage to define message schemas for data exchange. The consistency of data exchange specifications may be effectively supported by implementing four types of testing of XML Schema usage in message schema definitions:

- The message schema conformance to the **XML Schema specification grammar**
- The message schema conformance to the **XML Schema specification semantics**
- The message schema conformance to **design quality** testing
- The **canonical semantics** testing of the message schema

For brevity, we refer to these four categories as schema grammar, schema semantics, schema design quality, and content semantics conformance category. Each of the conformance categories may be viewed as a separate test suite and is described in subsequent subsections.

2.1. Schema Grammar Conformance

The schema grammar conformance means that the message schemas developed by any integration subgroup must conform to the World Wide Web Consortium (W3C) XML schema specification. The conformance of this type can be performed by executing an ‘XML validating parser’ against the published W3C XML Schema specification such as Xerces [Apache 2003] and MSXML [Microsoft 2003]. Alternatively, the W3C online XML schema validator is also available at http://apps.gotdotnet.com/xmltools/xsdvalidator/. The National Institute of Standards and

Technology (NIST) also offers a set of test suites for conformance testing of XML validating parsers to help select a parser of choice [NIST3].

An XML message schema conforming to the schema grammar only ensures that the data structure definitions defined in the schema are computer interpretable. In effect, the message schema can be used to validate an XML instance document (an actual message) as to whether its content conforms to the intended data structures. However, this conformance type does not ensure that the data structure definitions are logically consistent and unambiguous.

2.2. Schema Semantics Conformance

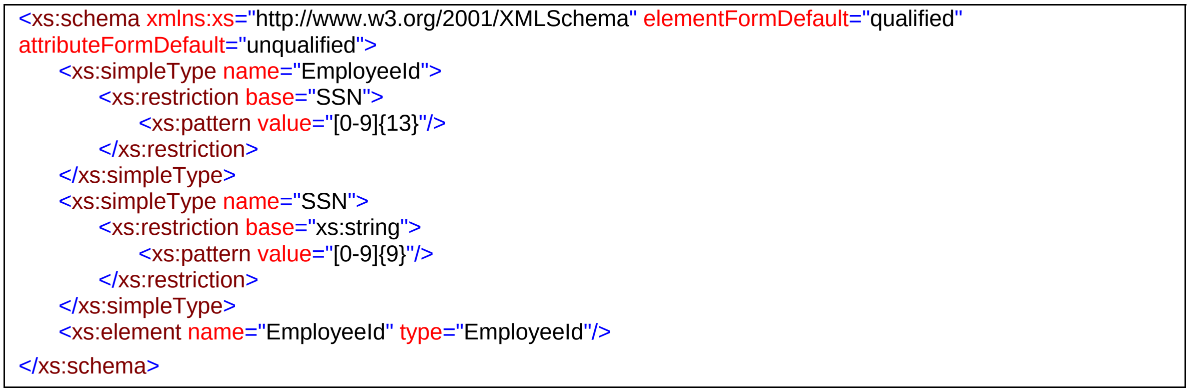
The schema semantics conformance means that the candidate XML schema defines a semantically valid model with respect to the standard W3C XML Schema semantics. In other words, we must ensure that the candidate XML schema does not define a conflicting relationship (see Figure 1) or an ambiguous information model (see Figure 2). For this conformance category, the IBM Schema Quality Checker [IBM 2003] offers the testing functionality.

Figure 1 shows a conflicting relationship between `SSN` and `EmployeeId`. First, the `SSN` type is defined as a 9-digit numerical string. The `EmployeeId` is the employee's `SSN` appended with a four-digit suffix. This is a grammatically valid XML schema, but one cannot create any valid `EmployeeId` element. This is because a string cannot be valid for both restrictions. The semantics associated with the restrictions on different facets are treated as a conjunction. Consequently, the `EmployeeId` must be valid for both facets.

Figure 2 shows an ambiguous information model. The `SSN` type is defined the same way as in Figure 1. For the same reason as above, the schema designer defines the `EmployeeId` type based on the `SSN` type. However, she uses a pattern facet to restrict its value to be a 13-digit numerical string. This schema is also grammatically valid; however, the result is that both 9- and 13-digit numerical strings are valid `EmployeeId`'s. That is because these same facets are interpreted as alternatives (disjunction).

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
attributeFormDefault="unqualified">
    <xs:simpleType name="EmployeeId">
        <xs:restriction base="SSN">
            <xs:length value="13"/>
        </xs:restriction>
    </xs:simpleType>
    <xs:simpleType name="SSN">
        <xs:restriction base="xs:string">
            <xs:pattern value="[0-9]{9}"/>
        </xs:restriction>
    </xs:simpleType>
    <xs:element name="EmployeeId" type="EmployeeId"/>
</xs:schema>
```

Figure 1: A Conflicting Schema Example.



```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
attributeFormDefault="unqualified">
    <xs:simpleType name="EmployeeId">
        <xs:restriction base="SSN">
            <xs:pattern value="[0-9]{13}"/>
        </xs:restriction>
    </xs:simpleType>
    <xs:simpleType name="SSN">
        <xs:restriction base="xs:string">
            <xs:pattern value="[0-9]{9}"/>
        </xs:restriction>
    </xs:simpleType>
    <xs:element name="EmployeeId" type="EmployeeId"/>
</xs:schema>
```

Figure 2: An Ambiguous Schema Example.

2.3. Schema Design Quality

The schema design quality conformance means that the designed schema complies with some set of best practices and organizational-specific requirements. The best practice rules may be drawn from experienced system integrators and/or XML architects. Tests within this category seek to enhance the usability/re-usability and interoperability support of the schema such as the schema's ability to capture and enforce desired semantics, extensibility, ease of maintenance, and implementation and processing efficiency.

A tool to test for conformance to such requirements must capture design rules and expertise in an executable knowledge base. Some design rules are generic, others are organizationally specific or architecturally dependent. We have developed a **framework** that allows such knowledge to be identified, collected, and used for analysis of schema design quality. We describe this framework in Section 3.

2.4. Content Semantics Conformance

Content semantics conformance means that a canonical semantic model of data exchange specifications is maintained and used across the enterprise. Tools to verify this conformance must deal with an evolving semantic model. They must be able to identify semantic overlaps and duplicates between a newly created message schema and the base set of message schemas (served as the canonical semantic model). When an overlap or a duplicate is identified, the tools must compute a similarity measure and suggest a strategy for reconciliation. Once the new schema is verified, it can be included as part of the base schemas. As the number of schemas grows, the time and effort requiring the coordination authority team to ensure the canonical semantic model grow. Such tools would make this recursive task more efficient.

Creating tools for this conformance category is a difficult problem but research results have shown promising approaches. Stuckenschmidt and Visser (2000) described three possible similarity measures of various complexity and robustness including one based on the Rough set theory [Pawlak 1982], another using the Bayesian theorem [Bayes 1963], and the other based on the Fuzzy set theory [Zadeh 1965]. Peng *et al.* (2002) also detailed a Bayesian approach to measure the semantic similarity. Ambite and Knoblock

(1995) introduced an approach to reconcile schemas to accommodate discrepancies between a document instance and its schema.

These approaches should not be viewed as competing but as complementary at this early stage of content semantics testing development. While traditional applications of these approaches are in the search algorithms and database integration areas, our research objective is to bring these approaches together and fine-tune them to yield maximum benefits to the large and evolving enterprise integration project.

3. Schema Design Quality Testing Approach

The knowledge about the schema design quality is typically obtained from an XML designer expert knowledge or from organizational conventions (some of which may be adopted from standard conventions). The list below describes some design quality knowledge collected within our initial research project.

- **Test for non-determinism** [OAG]. This includes data structures that fall into "type by attribute" category, meaning that type is hidden in the attribute. As an example, consider defining a `Party` element with a `type` or `qualifier` attribute. Doing so unnecessarily limits the extensibility of the expression because types are hidden. Separate types should be defined with relationships – in this case, we should define `ShipToParty` and `ShipFromParty` as subtypes of `Party`. Then, one can associate any unique property to the two subtypes. In addition, in any place where only the `ShipToParty` type is appropriate, it can be explicitly indicated in the model and validated by a parser. This constraint cannot be expressed with the XML Schema when the type is hidden in the attribute.
- **Test for correct use of the upper camel case for long tag names**. Upper camel case tags should be parsed to spell-check each sub-string to make sure it is a valid word or abbreviation. An all upper case substring may be recognized as a specific acronym and ignored, or checked against a list of allowable acronyms. Long tag names have become a convention because of increased computing power. The upper camel case convention is also adopted as a standard for such purpose (ISO 11179 provides a guideline for the upper camel case convention [ISO 1999]). This test ensures common usage of the upper camel case across an organization and facilitates content-semantics testing as well.
- **Test for extensibility and reusability of the schema**. This test can be done by, for example, praising the use of global types and warning against the use of anonymous types whose content model is locally defined within an element. Globally defined types allow reuse while the anonymous types do not. In addition, anonymous types make it harder to identify similarity.
- **Test for the use of weak typing** [OAG]. Contents within complex structured elements that are typed as an XML Schema primitive Data Type are regarded as weak typing. Weak typing provides little value and semantics to validation and application processing. In addition, it decreases the schema extensibility and lowers the efficiency of content semantics testing.

- **Test for compliance with other design principles**. Although the XML schema standard provides a large number of features to specify information structure and semantics, some features are not appropriate for integration projects. In addition, some of them must be used with caution. Two of these concerns are described below.
    - *Ease of maintenance.* For example, the derivation based on restriction is not recommended because it is prone to the kinds of inconsistency shown in Figure 1 and Figure 2. Furthermore, the restriction of complex types has to restate all the contents of the base type (so called feature regression); hence, changes in the base type require update to all derived types.
    - *Schema clarity.* The schema clarity makes the schema easier to understand and implement. For example, use of default namespace for an imported schema is not recommended and all elements and types should have associated namespaces. One consequence of this is that a schema should not be defined with 'no target namespace'.
- **Test the schema for its ability to facilitate the use of an existing standard in the instance.** This test may be done through cross-referencing concepts that refer to existing content standards. Such concepts should be structured so that their attributes allow specification of metadata (pointing to associated standards) and/or they should be typed based on enumerations.
- **Test for organizational specific requirements.** Some of the issues that may be tested in this sub-category include specific target namespaces, namespace abbreviations, and consistent use of qualified or non-qualified elements and attribute forms.

These are small examples of design principles that may be encoded as executable test rules within the schema design quality conformance category. In an on-going project, we research and document extensively best practices from a number of XML design guidance documents such as the Air Force Global Combat and Support System XML Guideline document [GCSS], ASC X12 Reference Model for XML Design [X12], Korean Institute for Electronic Commerce XML Guideline [KIEC], and more. This knowledge must be captured in a computer interpretable format so that it can be used by the coordination authority team. Sometimes the knowledge can be encoded as a set of rules; other times it requires sophisticated routines to capture heuristics and to connect to other knowledge sources. Finally, since knowledge can evolve, each test should be a completely self-contained executable module.

4. Implementation of Schema Design Quality Testing

The schema design quality module will be built within the B2B Testbed context [NIST2], which is designed to be a neutral and persistent environment that provides reusability, accumulation of organizational knowledge and lessons learned, coordination, and cost sharing among industrial participants. Since some test cases can be organizational specific, the test tool will be designed as a knowledge repository. Each user can create a profile and add test cases incrementally to the profile. The tool will be a web-based client/server application where execution occurs on the server side; hence, technologies used to encode and execute the knowledge will be transparent to the user. However, in order to keep the knowledge open, the knowledge representation will be declarative. For that reason, simple knowledge will be encoded using W3C XPATH/XSLT expression [XPATH, XSLT] using the Schematron schema [Schematron] and the complex knowledge will be encoded using rule-based languages such as the Java-based Expert System Shell script [JESS]. The expert system shell script, which runs as a server side application, allows the server to connect to multiple knowledge sources. In the long term, our goal is for the users to be able to submit additional knowledge and conformance rules. Figure 3 shows the implementation architecture.

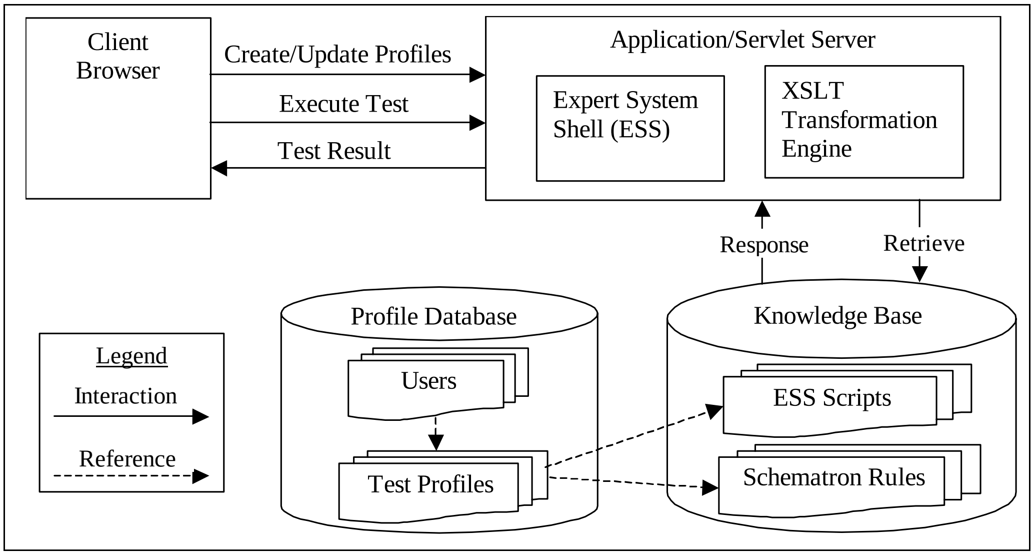


Figure 3: An Implementation Architecture for Schema Design Quality Testing.

```
<pattern name="Use of anonymous type.">
    <rule context="xs:element">
        <report test="xs:complexType">Use of anonymous type is not recommended for extensibility reason. It is
recommended that global type be defined and the element is declared based on that global type.
        </report>
        <report test="xs:simpleType">Use of anonymous type is not recommended for extensibility reason. It is
recommended that global type be defined and the element is declared based on that global type.
        </report>
    </rule>
</pattern>
```

Figure 4: A Schematron Snippet Detecting the Use of Anonymous Type.

As an example, we show a snippet of Schematron code capturing the rule for detecting the use of anonymous type in Figure 4. The snippet looks for a pattern, in which the `xs:complexType` or `xs:simpleType` appears as a child of the `xs:element`, an XML schema. If such a pattern is found, it prints out the corresponding warning messages to the user. Snippets like these are declarative and are self-contained knowledge modules. It can be stored and executed independently in the knowledge base.

5. Conclusion

The paper presented a framework for managing an evolving integration project, which utilizes XML as an integration medium. As the project evolves so do the message schemas used for enterprise integration and using this proposed framework to help carefully manage the evolution of schemas allows software components built on those schemas to be reusable. This eliminates the need for and cost of point-to-point integrations. In addition, the framework helps ensure that integration subprojects adhere to the same structure and representation when they overlap in functionalities and semantics. These enhancements provided by the proposed framework collectively may significantly promote interoperability among applications. Moreover, the framework is not only applicable to the integration projects within an enterprise, but also to the integration projects within a supply chain where data exchange specifications also evolve. Likewise, international standards such as the ebXML Core Components [EbCC], which envision a growing repository of semantics, could use this framework to maintain a canonical and interoperable semantic model.

6. Disclaimer

Certain commercial software products are identified in this paper. These products were used only for demonstration purposes. This use does not imply approval or endorsement by NIST, nor does it imply that these products are necessarily the best available for the purpose.

7. References

[Ambite and Knoblock 1995] Ambite, J.L. and Knoblock, C.A., Reconciling distributed information sources. *In Working Notes of the AAAI Spring Symposium on Information Gathering in Distributed Heterogeneous Environments*, Palo Alto, CA (1995).

[Bayes 1963] Bayes, T., *An essay towards solving a problem in the doctrine of chance*. Phil. Trans. Reproduced in: W.E. Deming and Haner (eds.) New York (1963).

[EbCC] DISA UN/CEFACT. *EbXML Core Component Specification version 1.9* (December 2002). Available online via < http://webster.disa.org/cefact-groups/tmg/downloads/CCWG/for_review/CCTS_V_1pt90.zip>.

[GCSS] Destefani, C., *Global Combat Support System - Air Force BOD Development Process presentation slides*. Open Application Group Meeting, New Orleans, LA (October 2002). Available online via < http://openapplications.org/downloads/meetings/200210%20NewOrleans/02-October New-Orleans.zip>.

[ISO 1999] International Organization for Standardization. *ISO/IEC 11179-1 Specification and standardization of data element – Part 1: Framework* (1999).

[JESS] Friedman-Hill, E., *Jess the Rule Engine for the Java^{TM} Platform Version 6.0a8*. Internet web site available online via <http://herzberg.ca.sandia.gov/jess/> (accessed July 2002).

[KIEC] Korean Institute for Electronic Commerce. *Guidelines for Development of XML Electronic Messages in Korea* (March 2003). Available online via <www.xeni.co.kr/support/KIECGuidelineFinal_english_.pdf>.

[Linthicum 1999] Linthicum, D.S., *Enterprise Application Integration*, Pearson Education (1999).

[NIST1] National Institute of Standards and Technology. *Interoperability Cost Analysis in the Automotive Supply Chain* (March 1999). Available online via <http://www.mel.nist.gov/msid/sima/interop_costs.pdf>.

[NIST2] National Institute of Standards and Technology. *The B2B Interoperability Testbed*. Internet web site available online via <http://www.mel.nist.gov/msid/oagnisttestbed/> (accessed January 2004).

[NIST3] National Institute of Standards and Technology. *XML Schema Conformance Test Suite*. Available online via <http://xw2k.sdct.itl.nist.gov/brady/xml/generate.asp?tech=XML_Schema>. Accessed August 2003.

[OAG] Rowell, M. and Feblowitz, M., *OAGIS 8.0 Design Document*, Open Application Groups (2002).

[Pawlak 1982] Pawlak, J., Rough Sets, *International Journal of Information and Computers*, 11, pp.341-356 (1982).

[Peng et al. 2002] Peng, Y., Zou, Y., Luan, X., Ivezic, N., Gruninger, M., and Jones, A., Towards semantic-based integration for e-business, *International Symposium on manufacturing and Applications*, Orlando, FL (June 2002).

[Schematron] Jelliffe, R., *The Schematron Assertion Language 1.5*, Academia Sinica Computing Center. Internet web site available online via <http://www.ascc.net/xml/resource/schematron/Schematron2000.html> (accessed August 2003).

[Stuckenschmidt 2000] Stuckenschmidt, H. and Visser, U., Semantic translation based on approximate re-classification. *Proceedings of the Workshop "Semantic Approximation, Granularity and Vagueness, KR'00* (2000).

[W3C 2001] World Wide Web Consortium (May 2001). *XML Schema Part 1: Structure W3C Recommendation*. Available online via <http://www.w3.org/TR/xmlschema-1>.

[Apache 2003] The Apache Software foundataion. *Xerces2 Java Parser*. Available on line via <http://xml.apache.org/xerces2-j/index.html>. Accessed November 2003.

[Microsoft 2003] Microsoft Corporation. *Microsoft XML Parser*. Available online via <http://msdn.microsoft.com/library/default.asp?url=/downloads/list/xmlgeneral.asp>. Accessed November 2003.

[IBM 2003] IBM Corporation. *The IBM Schema Quality Checker*. Available online via <http://www.alphaworks.ibm.com/tech/xmlsqc>. Accessed August 2003.

[X12] ASC X12C Communications and Controls Subcommittee. *ASC X12 Reference Model for XML Design* (October 2002), Technical Report Type II, X12C/2002-61.

Available online via <http://www.x12.org/x12org/xmldesign/X12Reference_Model_For_XML_Design.pdf>.

[XPATH] World Wide Web Consortium. *XML PATH Language Version 1.0* (November 1999). Available online via <http://www.w3.org/TR/xpath>.

[XSLT] World Wide Web Consortium. *XSL Transformations (XSLT) Version 1.0* (November 1999). Available online via <http://www.w3.org/TR/xslt>.

[Zadeh 1965] Zadeh, L., Fuzzy sets. *Information and Control*, pp.338 - 353 (1965).